



BANNARI AMMAN INSTITUTE OF TECHNOLOGY

An Autonomous Institution Affiliated to Anna University Chennai - Approved by AICTE - Accredited by NAAC with "A" Grade

SATHYAMANGALAM - 638 401 ERODE DISTRICT TAMIL NADU INDIA

Ph: 04295-226000 / 221289 Fax: 04295-226666 E-mail: stayahead@bitsathy.ac.in Web: www.bitsathy.ac.in

ONE CREDIT ASSIGNMENT REPORT

ONE CREDIT COURSE ASSIGNMENT REPORT

Submitted by
ANOOSKAVIN G
(191CS119)
Final year - CSE



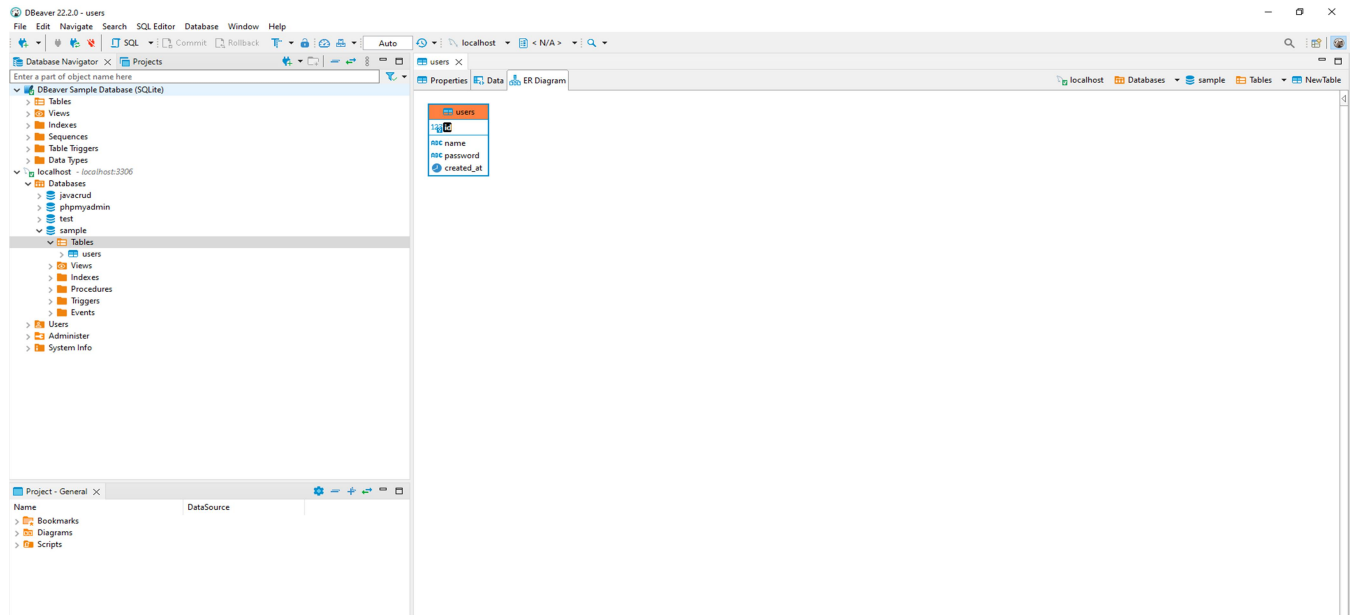
BANNARI AMMAN INSTITUTE OF TECHNOLOGY
(An Autonomous Institution Affiliated to Anna University, Chennai)
SATHYAMANGALAM-638401

September 2022

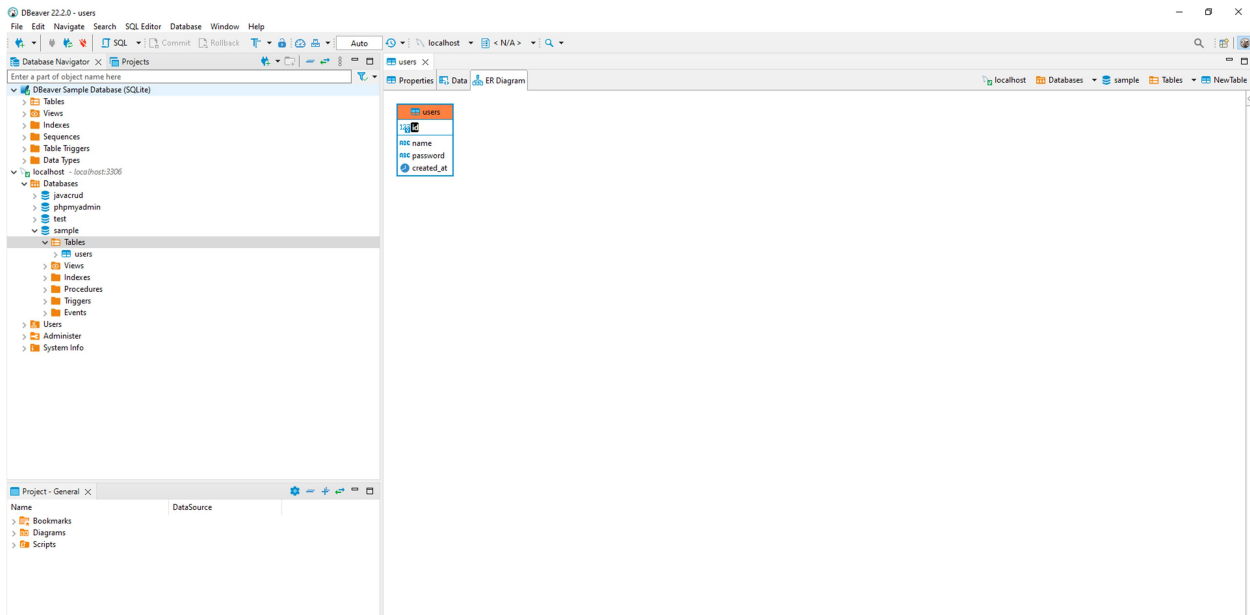
| | |
|------------------|---|
| Ex.no: 1 | Create a Table with 4 columns with primary key and time stamp data-type |
| Date: 24.09.2022 | |

Implementation Steps & Output:

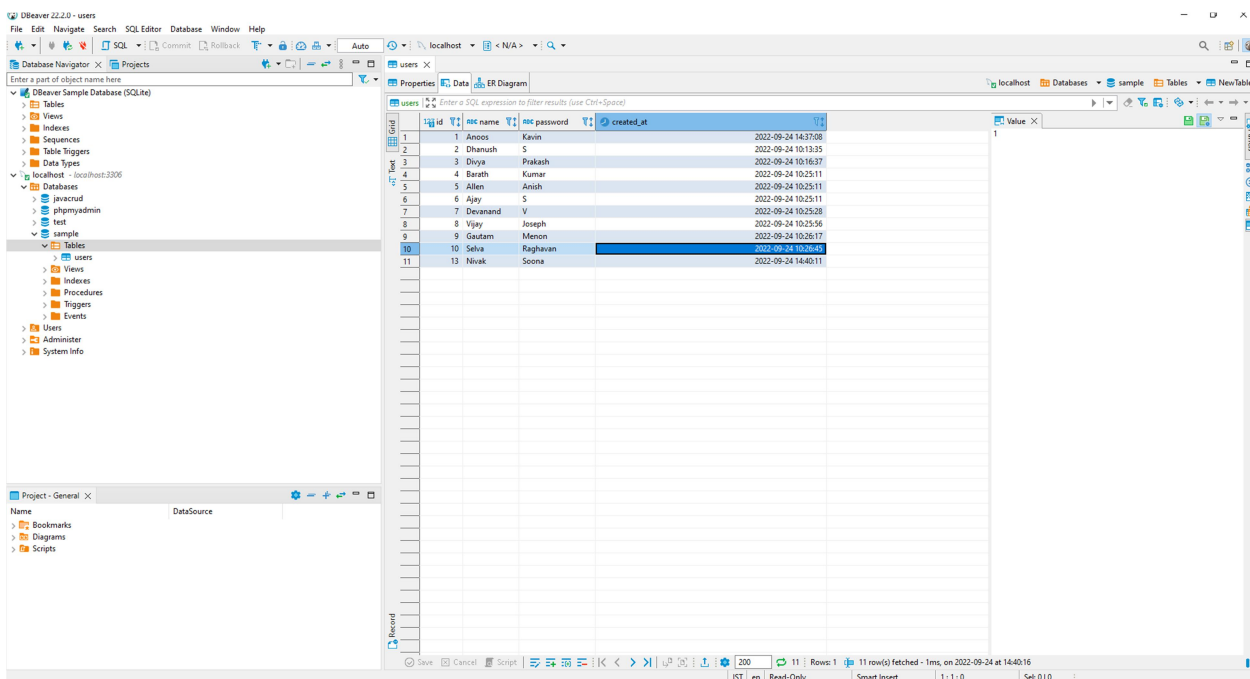
1. Open **DBeaver**. Create **MySql** database in our DBeaver.
2. Put the credentials for DB and connect it.
3. Create a table by right clicking the database and name it.
4. Create the columns for the table you created.



5. Enter the constraints for the columns like **Primary key, Not null etc.,**



6. Add the values for the table.



We can also create the following table using **Queries** given below,

```
CREATE DATABASE sample;
CREATE TABLE users(
  id INT NOT NULL AUTO INCREMENT,
  name VARCHAR(100) NOT NULL,
  password VARCHAR(100) NOT NULL,
  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL,
  PRIMARY KEY(id)
);
```

| | |
|------------------|--|
| Ex.no: 2 | Write All API methods with query using POSTMAN |
| Date: 24.09.2022 | |

Implementation Steps & Output:

1. Write the code for the required packages to use it such as **mysql,body-parser,cors,express**

```
var expres = require('express')
var app = expres()
var bodyparser = require('body-parser')
var mysql = require('mysql')
var cors = require('cors')

app.use(cors())
app.use(bodyparser.json())

app.use(bodyparser.urlencoded({
  extended : true
}));
```

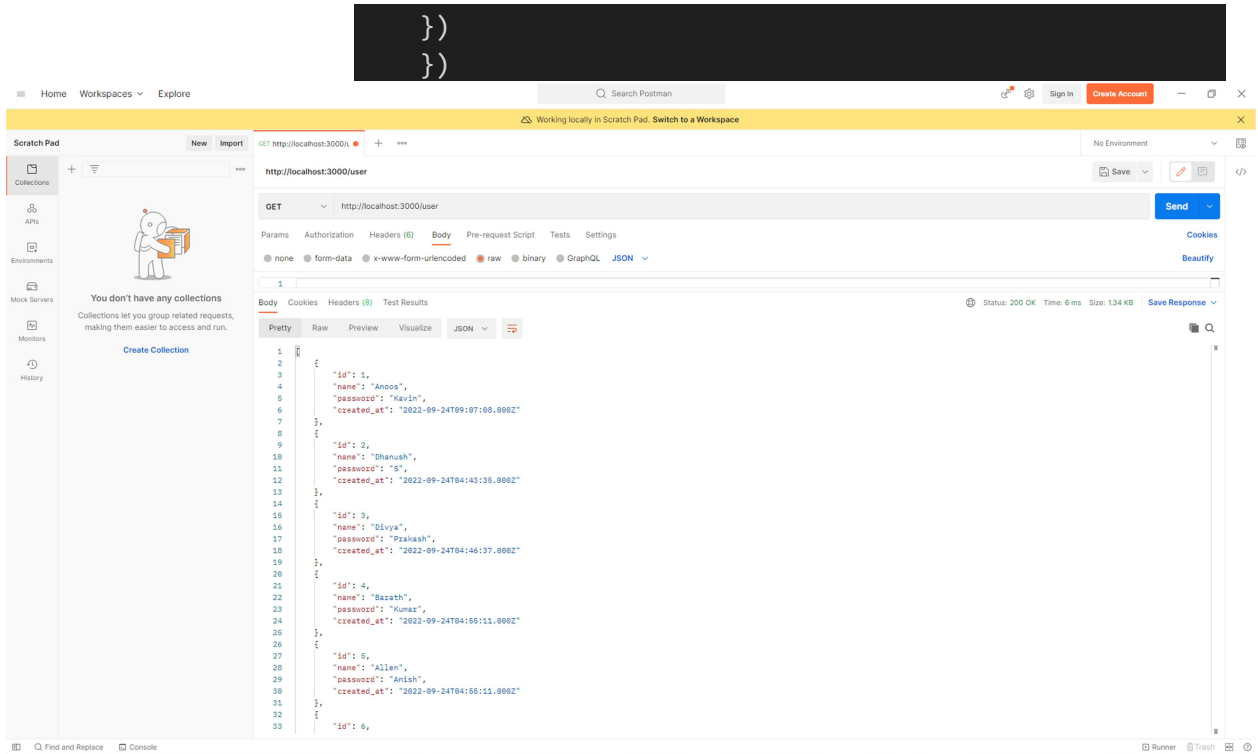
2. Code for connecting MySQL DB.

```
var dbConn = mysql.createConnection({
  host: 'localhost',
  user: 'root',
  password: '',
  database: 'sample'
})
dbConn.connect()
```

3. GET METHOD:

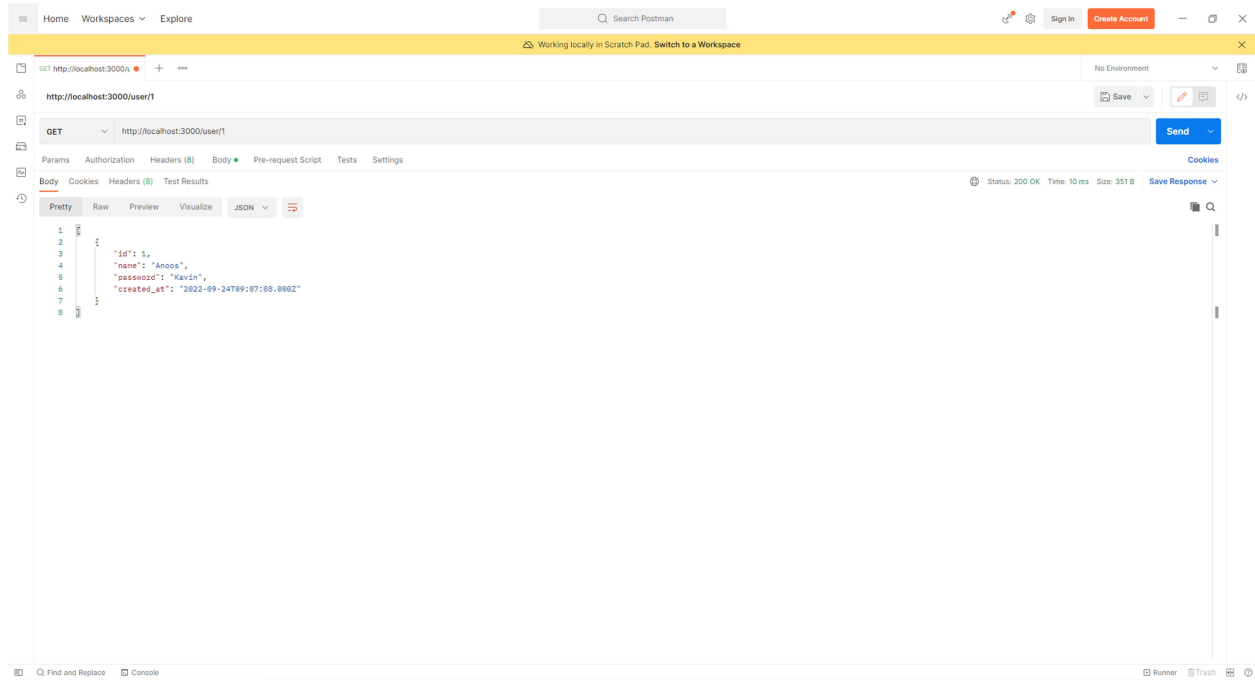
- Obtain all the values of the table using GET method.

```
app.get('/user',function(req,res){
  dbConn.query('select * from
users',function(error,results,fields){
    if(error) throw error;
    return res.send(results);
  });
});
```



- Obtain the details for the particular id using GET method

```
app.get('/user/:id',function(req,res){
  let id = req.params.id;
  console.log(id);
  dbConn.query(`select * from users where id = ${id}`
  ,function(error,results,fields){
    if(error) throw error;
    return res.send(results);
  })
})
```



4. POST METHOD:

- Create a user by passing values via **body**

```
app.post('/user',function(req,res){  
    let name = req.body.name;  
    let pwd = req.body.pwd;  
    console.log(name,pwd);  
    dbConn.query(`insert into users(name,password)  
values('${name}','${pwd}')`,function(error,results,fields){  
        if(error) throw error;  
        return res.send(results);  
    })  
})
```

The screenshot displays two applications side-by-side. On the left is DBeaver, showing a table named 'users' in a database. The table has columns: id, name, password, and created_at. The data includes users like Anous, Dhanush, Divya, Barath, Allen, Ajay, Devanand, Vijay, Gaudam, Sathya, Nivak, and Bhemaia. On the right is Postman, showing a POST request to 'http://localhost:3000/user'. The request body is a JSON object: { "name": "Ajayak", "pwd": "Bheemaa" }. The response is a JSON object: { "fieldCount": 0, "affectedRows": 1, "insertId": 16, "serverStatus": 2, "warningCount": 0, "message": "", "protocol41": true, "changedRows": 0 }.

| id | name | password | created_at |
|----|----------|----------|---------------------|
| 1 | Anous | Kavin | 2022-09-24 10:13:35 |
| 2 | Dhanush | S | 2022-09-24 10:13:35 |
| 3 | Divya | Prakash | 2022-09-24 10:16:37 |
| 4 | Barath | Kumar | 2022-09-24 10:25:11 |
| 5 | Allen | Anish | 2022-09-24 10:25:11 |
| 6 | Ajay | S | 2022-09-24 10:25:11 |
| 7 | Devanand | V | 2022-09-24 10:25:28 |
| 8 | Vijay | Joseph | 2022-09-24 10:25:56 |
| 9 | Gaudam | Menon | 2022-09-24 10:26:17 |
| 10 | Sathya | Raghavan | 2022-09-24 10:26:45 |
| 11 | Nivak | Soona | 2022-09-24 14:40:11 |
| 12 | Ajayak | Bheemaa | 2022-09-24 15:17:13 |

- Create a user by passing values via **Params**

```
app.post('/user/:name/:pwd',function(req,res){
    let name = req.params.name;
    let pwd = req.params.pwd;
    console.log(name,pwd);
    dbConn.query(`insert into users(name,password)
values('${name}','${pwd}')`,function(error,results,fields){
        if(error) throw error;
        return res.send(results);
    })
})
```


The screenshot displays two applications side-by-side. On the left is DBeaver 22.0.9, showing a table named 'users' with columns 'id', 'name', 'password', and 'created_at'. The table contains 12 rows of user data. On the right is Postman, showing a PUT request to 'http://localhost:3000/user/John/Peter'. The request body is a JSON object: { "id": 10, "name": "Raghavan", "password": "123456", "created_at": "2022-09-24 15:23:52" }. The status is 200 OK.

| id | name | password | created_at |
|----|----------|----------|---------------------|
| 1 | Anoos | Kevin | 2022-09-24 14:37:08 |
| 2 | DHANUSH | S | 2022-09-24 15:21:32 |
| 3 | Dhoya | Prakash | 2022-09-24 15:16:37 |
| 4 | Ravath | Kumar | 2022-09-24 10:25:11 |
| 5 | Allen | Anish | 2022-09-24 10:25:11 |
| 6 | Ajey | S | 2022-09-24 10:25:11 |
| 7 | Devanand | V | 2022-09-24 10:25:38 |
| 8 | Vijay | Joseph | 2022-09-24 10:25:36 |
| 9 | Gaudam | Menon | 2022-09-24 10:26:17 |
| 10 | Selva | Raghavan | 2022-09-24 10:26:45 |
| 11 | Nivak | Soona | 2022-09-24 14:40:11 |
| 12 | John | Peter | 2022-09-24 15:23:52 |

5. PUT METHOD

- Update the user name with respect to id by using PUT method.

```
app.put('/user/:id/:name',function(req,res){
  let id = req.params.id;
  let name = req.params.name;
  console.log(id);
  dbConn.query(`update users set
name='${name}' where id = ${id}
`,function(error,results,fields){
    if(error) throw error;
    return res.send(results);
  })
})
```

The screenshot displays the DBaiver 22.2.0 interface. On the left, the 'Database Navigator' shows a project named 'sample' with a table named 'users'. The 'users' table is selected, and its data is shown in the 'Data' tab. The table has columns: id, name, email, password, and created_at. The data is as follows:

| id | name | email | password | created_at |
|----|----------|----------|----------|---------------------|
| 1 | Anoos | Kevin | | 2022-09-24 14:37:08 |
| 2 | DHANUSH | S | | 2022-09-24 15:21:32 |
| 3 | Dhruv | Prakash | | 2022-09-24 15:16:37 |
| 4 | Ravith | Kumar | | 2022-09-24 10:25:11 |
| 5 | Allen | Anish | | 2022-09-24 10:25:11 |
| 6 | Ajay | S | | 2022-09-24 10:25:11 |
| 7 | Devanand | V | | 2022-09-24 10:25:38 |
| 8 | Vijay | Joseph | | 2022-09-24 10:25:36 |
| 9 | Gaudam | Menon | | 2022-09-24 10:26:17 |
| 10 | Selva | Raghavan | | 2022-09-24 10:26:45 |
| 11 | Nivak | Devine | | 2022-09-24 14:40:11 |

On the right, the REST client shows a PUT request to 'http://localhost:3000/user/2/DHANUSH'. The response is a JSON object:

```
{
  "statusCode": 200,
  "message": "User updated successfully",
  "affectedRows": 1,
  "insertId": 0,
  "serverStatus": 2,
  "warningCount": 0,
  "protocol41": true,
  "changeRows": 1
}
```

6. DELETE METHOD

- Delete a user with respect to id by using DELETE method.

```
app.delete('/user/:id',function(req,res){
  let id = req.params.id;
  console.log(id);
  dbConn.query(`delete from users where id = ${id}`,function(error,results,fields){
    if(error) throw error;
    return res.send(results);
  })
})
```

The screenshot displays two applications side-by-side. On the left is DBeaver 22.0.9, showing a connection to a MySQL database named 'sample' on 'localhost'. The 'users' table is selected, showing a list of users with columns: id, first_name, last_name, email, phone_number, hire_date, job_id, salary, commission_pct, manager_id, and department_id. On the right is Postman, showing a DELETE request to 'http://localhost:3000/user/16'. The response is a JSON object with status 200 OK, indicating a successful deletion.

| id | first_name | last_name | email | phone_number | hire_date | job_id | salary | commission_pct | manager_id | department_id |
|----|------------|-----------|-------|--------------|---------------------|--------|--------|----------------|------------|---------------|
| 1 | Anoos | Kavin | | | 2022-09-24 14:37:08 | | | | | |
| 2 | Dhanush | S | | | 2022-09-24 10:13:35 | | | | | |
| 3 | Dhoya | Prakash | | | 2022-09-24 10:16:37 | | | | | |
| 4 | Sarath | Kumar | | | 2022-09-24 10:25:11 | | | | | |
| 5 | Allen | Anish | | | 2022-09-24 10:25:11 | | | | | |
| 6 | Ajey | S | | | 2022-09-24 10:25:11 | | | | | |
| 7 | Devanand | V | | | 2022-09-24 10:25:38 | | | | | |
| 8 | Vijay | Joseph | | | 2022-09-24 10:25:36 | | | | | |
| 9 | Gaudam | Menon | | | 2022-09-24 10:26:17 | | | | | |
| 10 | Selva | Raghavan | | | 2022-09-24 10:26:45 | | | | | |
| 11 | Isaac | Seena | | | 2022-09-24 14:40:11 | | | | | |

Full Code:

```
var expres = require('express')
var app = expres()
var bodyparser = require('body-parser')
var mysql = require('mysql')
var cors = require('cors')

app.use(cors())
app.use(bodyparser.json())

app.use(bodyparser.urlencoded({
  extended : true
}));
var dbConn = mysql.createConnection({
  host: 'localhost',
  user: 'root',
  password: '',
  database: 'sample'
})
dbConn.connect()

app.get('/user',function(req,res){
  dbConn.query('select * from users',function(error,results,fields){
```

```
        if(error) throw error;
        return res.send(results);
    })
})

app.post('/user',function(req,res){
    let name = req.body.name;
    let pwd = req.body.pwd;
    console.log(name,pwd);
    dbConn.query(`insert into users(name,password)
values('${name}','${pwd}')`,function(error,results,fields){
        if(error) throw error;
        return res.send(results);
    })
})

app.get('/user/:id',function(req,res){
    let id = req.params.id;
    console.log(id);
    dbConn.query(`select * from users where id = ${id}
`,function(error,results,fields){
        if(error) throw error;
        return res.send(results);
    })
})

app.delete('/user/:id',function(req,res){
    let id = req.params.id;
    console.log(id);
    dbConn.query(`delete from users where id =
${id}`,function(error,results,fields){
        if(error) throw error;
        return res.send(results);
    })
})

app.put('/user/:id/:name',function(req,res){
    let id = req.params.id;
    let name = req.params.name;
    console.log(id);
    dbConn.query(`update users set name='${name}' where id = ${id}
`,function(error,results,fields){
        if(error) throw error;
        return res.send(results);
    })
})
```

```
  })

  app.post('/user/:name/:pwd',function(req,res){
    let name = req.params.name;
    let pwd = req.params.pwd;
    console.log(name,pwd);
    dbConn.query(`insert into users(name,password)
values('${name}','${pwd}')`,function(error,results,fields){
      if(error) throw error;
      return res.send(results);
    })
  })

  app.listen(3000,function(){
    console.log("Server Started");
  })
```